arXiv:1301.3592v3 [cs.LG] 21 Feb 2013

Deep Learning for Detecting Robotic Grasps

Ian Lenz,[†] **Honglak Lee,**[*] **and Ashutosh Saxena.**[†]
[†] Computer Science Department, Cornell University.
[*] EECS, University of Michigan, Ann Arbor.
Email: ianlenz@cs.cornell.edu, honglak@eecs.umich.edu,asaxena@cs.cornell.edu

Abstract

In this work, we consider the problem of detecting robotic grasps in an RGB-D view of a scene containing objects. We present a two-step cascaded structure, where we have two deep networks, with the top detections from the first one re-evaluated by the second one. The first deep network has fewer features, is therefore faster to run and makes more mistakes. The second network has more features and gives better detections. Unlike previous works that need to design these features manually, deep learning gives us flexibility in designing such multi-step cascaded detectors, outperforming the previous state-of-the-art.

1 Introduction and Problem Statement

Robotic grasping is a challenging problem for many reasons including perception, planning and control. Some recent works [3, 15] address part of this problem by converting this problem into a detection problem, where given a noisy, partial view of the object from a RGBD camera, the goal is to infer the top locations of where the robotic gripper could be placed. Unlike generic vision problems based on static images, such robotics perception problems are often used in closed loop with the controllers, so there are stringent requirements on performance and computational speed. More importantly, roboticists do not have enough time to hand-engineer features for every robotic task (beyond grasping, e.g., door opening [2, 5]), thus making it crucial to learn features automatically.

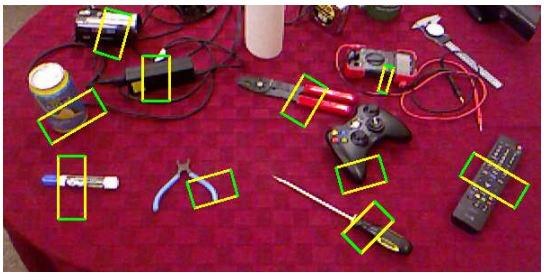


Figure 1: **Detecting robotic grasps.** – A cluttered lab scene labeled with rectangles corresponding to robotic grasps for objects in the scene. Green lines correspond to robotic gripper plates. We use a two-pass system based on deep learning to learn features and perform detection for robotic grasping.

There are two main contributions of our work. First, while deep learning methods have shown impressive results in many areas (e.g., [1, 4, 6, 7, 9, 10, 11, 16]), these approaches are generally applied in the context of *recognition*. Grasping is, inherently, a *detection* problem, and previous applications of deep learning to detection have typically focused on specific applications such as face detection [12]. Our goal is not only to infer a viable grasp, but to infer the optimal grasp for a given object, to give the robot the best chance of successfully grasping it. Second, to the best of our knowledge, our work is the first one that applies deep learning to the problem of robotic grasping.

In this work, we propose a two-step cascaded deep learning detection system, where we have fewer features in the first layer, giving us faster but only approximately accurate detections, and we have more features in the second layer, giving us more accurate detections. We feed the top rectangles from the first layer into the second layer. Unlike manually designed two-step features in [3], our method uses deep learning and hence it is easier to design detectors that not only give higher performance but are also computationally efficient.

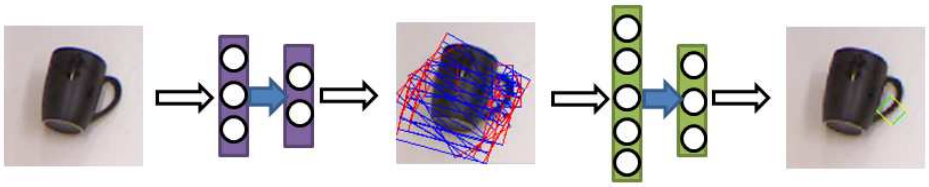


Figure 2: **Illustration of our two-stage detection process.** – Given an image of an object to grasp, a small deep network is used to exhaustively search potential rectangles, producing a small set of top-ranked rectangles. A larger deep network is then used to find the top-ranked rectangle from these candidates, producing a single optimal grasp for the given object.

2 Our Approach

2.1 Representation for Robotic Grasping

In order to represent robotic grasps, we will follow an approach similar to [3]. We will represent potential grasps using oriented rectangles in image space, with one pair of parallel edges corresponding to the robotic gripper.

In order to extract fixed-size input for a deep network based on a grasping rectangle, we downsample the area contained by the rectangle, preserving aspect ratio, so that its maximum dimension corresponds to the input size for our deep network. This downsampled rectangle is centered in the receptive field for the deep network. Regions outside the downsampled, centered rectangle are padded with zeros and not considered when computing the reconstruction penalty.

2.2 Deep Learning for Detection

Using a standard feature learning approach such as sparse auto-encoder [8], a deep network can be trained for the problem of grasping rectangle recognition, i.e., does a given rectangle in image space correspond to a valid robotic grasp? However, in a real-world robotic setting, our system needs to perform *detection*, i.e., given an image containing an object, where should the robot grasp the object? This task is significantly more challenging than simple recognition.

Two-pass Cascaded Detection. In order to perform detection, one naive approach could be to consider each possible oriented rectangle in the image (perhaps discretized to some level), and evaluate each rectangle with the deep network trained for recognition. However, such near-exhaustive search of possible rectangles (based on positions, sizes, and orientations) can be quite expensive in practice for real-time robotic grasping.

Motivated by multi-step cascaded approaches in previous work [3, 17], we take a two-pass approach to detection: First, we use a reduced feature set to determine a set of top candidates. Then, we use a larger, more robust feature set to rank these candidates.

However, these approaches require the design of two separate sets of features. In particular, it can be difficult to manually design a small set of first-stage features which is both quick to compute and robust enough to produce a good set of candidate detections for the second stage. However, using deep learning allows us to to circumvent the costly manual design of features by simply training networks of two different sizes, using the smaller for the exhaustive first pass, and the larger to re-rank the candidate detection results.

Preserving Aspect Ratio. It is important for grasping to preserve the aspect ratio while feeding it in the network. This is because the same object, if stretched, may no longer be graspable, as shown in Figure 3 Since the extra parts of rectangular receptive fields are padded with zero, our deep network exhibited a strong preference towards square grasping rectangles.

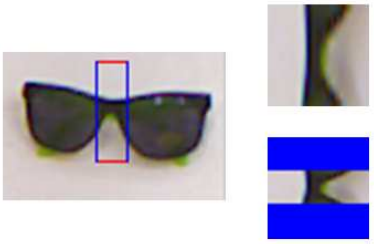


Figure 3: **Preserving aspect ratio.** – Left: a pair of sunglasses with a potential grasping rectangle. Red edges indicate gripper plates. Right-top: image taken from the rectangle and rescaled to fit a square aspect ratio. Right-bottom: same image, padded and centered in the receptive field. Blue areas indicate masked-out padding. When rescaled, the rectangle incorrectly appears graspable. Preserving aspect ratio and padding allows the rectangle to correctly appear non-graspable.

Figure 4: **Improvement from mask-based scaling.** – Left: Result without mask-based scaling. Right: Result with mask-based scaling.

In order to address this problem, we define a scaling factor for each visible unit, based on the inverse of the fraction of each data modality which remains masked in:

$$\Psi_s^{(t)} = ||s||_1 / \left(\sum_{i=1}^{N} s_i \mu_i^{(t)} \right) \quad (1)$$

where s is a binary vector indicating membership of each visible unit x_i in a particular modality, such as depth or image intensity. $\mu_i^{(t)}$ is 1 if $x_i^{(t)}$ is masked in, 0 otherwise.

When pretraining, we apply this scale to the visible features used for input and the squared reconstruction penalty for each case, but not to the ground-truth value used to compute reconstruction error.

3 Experiments

Figure 5: **Example objects from the Cornell grasping dataset.** – [3]. This dataset contains objects from a large variety of categories, and is available at `http://pr.cs.cornell.edu/grasping`.

Dataset.
We used the extended version of the Cornell grasping dataset [3] for our experiments. This dataset contains 1035 images of 280 objects, each annotated with several ground-truth positive and negative rectangles. While the vast majority of possible rectangles for most objects will be non-graspable, the dataset contains roughly equal numbers of graspable and non-graspable rectangles. We will show that this is useful for an unsupervised learning algorithm, as it allows learning a good representation for graspable rectangles even from unlabeled data.

We performed five-fold cross-validation, and present results for splits on a per image (i.e., the training set and the validation set do not share the same image) and per object (i.e., the training set and the validation set do not share any images from the same object).

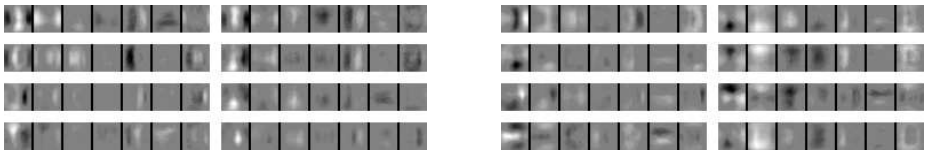
We take seven channels as input: YUV channels in the color space, depths, and the XYZ components of computed surface normals. With an image patch size of 24x24 pixels, we have 4032 (=24*24*7) input features. We trained a deep network with 200 hidden units each at the first and second layers and trained a logistic classifier at the top layer using our feature learning algorithm as described in the previous section, followed by supervised back-propagation.

Baselines.
We compare our recognition results in the Cornell grasping dataset with the features from [3], as well as the combination of these features and Fast Point Feature Histogram (FPFH) features [13]. We used a linear SVM for classification, which gave best results among all other kernels.

Metrics for Detection.
For detection, we compare the top-ranked rectangle for each method with the set of ground-truth rectangles for each image. We present results using two metrics, the "point" and "rectangle" metric.

For the point metric, similar to [14], we compute the center point of the predicted rectangle, and consider the grasp a success if it is within some distance from at least one ground-truth rectangle center. We note that this metric ignores grasp orientation, and therefore might overestimate the performance of an algorithm for robotic applications.

For the rectangle metric, similar to [3], let G be the top-ranked grasping rectangle predicted by the algorithm, and G^* be a ground-truth rectangle. Any rectangles with an orientation error of more than 30^o from G are rejected. From the remaining set, we use the common bounding box evaluation metric of intersection divided by union - i.e. $Area(G \cap G^*)/Area(G \cup G^*)$. Since a ground-truth rectangle can define a large space of graspable rectangles (e.g., covering the entire length of a pen), we consider a prediction to be correct if it scores at least 25% by this metric.



(a) Features corresponding to positive grasps (b) Features corresponding to negative grasps

Figure 6: **Features learned from grasping data** – Each feature contains seven channels - from left to right, depth, Y, U, and V image channels, and X, Y, and Z surface normal components. Left: 8 features with the strongest positive correlation to rectangle graspability. Right: similar, but strongest negative correlation.

4 Results and Discussion

4.1 Deep Learning for Robotic Grasp Detection

Figure 6 shows the features learned by the unsupervised phase of our algorithm which have a high correlation to positive and negative grasping cases. Figure 7 shows 3D meshes for the 4 features with the strongest positive and negative correlations to valid grasps. These were obtained using a standard meshing algorithm, with X and Y coordinates corresponding to positions in the deep network's receptive field, and Z coordinates corresponding to weight values to the depth channel for each location.

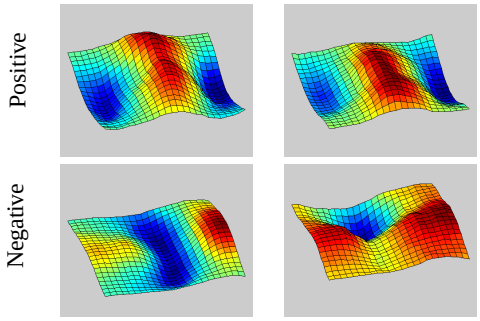


Figure 7: **3D depth features** – 3D meshes for depth channels of the 4 features with strongest positive (top) and negative(bottom) correlations to rectangle graspability. Feature shapes clearly correspond to graspable and non-graspable structures, respectively

Even *without any supervised information*, our algorithm was able to learn several features which correlate strongly to graspable cases and non-graspable cases. The first two positive-correlated features represent handles, or other cases with a raised region in the center, while the second two represent circular rims or handles. The negatively-correlated features represent obviously non-graspable cases, such as ridges perpendicular to the gripper plane and depressions between the gripper plates. From these features, we can see that even during unsupervised feature learning, our approach is able to learn a task-specific representation.

Table 1: **Recognition results for Cornell grasping rectangle dataset.** – Deep learning approaches significantly improve accuracy compared to baselines

Algorithm	Accuracy (%)
Jiang et al. [3]	84.7
Jiang et al. [3] + FPFH	89.6
Sparse AE	**93.7**

From Table 1, we see that the recognition performance is significantly improved with deep learning methods, improving 9% over the features from [3] and 4.1% over those features combined with FPFH features. Both L1 and group regularization performed similarly for recognition, but training separate first layer features decreased performance slightly.

Table 2 shows that, once mask-based scaling has been applied, all deep learning approaches except for training separate first-layer features outperform the hand-engineered features from [3] by up to 13% for the point metric and 17% for the rectangle metric, while also avoiding the need to design task-specific features.

Adaptability.
One important advantage of our detection system is that we can flexibly specify the constraints of the gripper in our detection system. Different robots have different grippers —PR2 has a wide gripper, while the Adept Viper arm has a smaller one. We can constrain the detectors to handle this. Figure 8

Table 2: **Detection results for point and rectangle metrics.** – deep learning, mask-based scaling, and the two-pass system all produce improvements

Algorithm	Image-wise split		Object-wise split	
	Point (%)	Rect. (%)	Point (%)	Rect. (%)
Jiang et al. [3]	75.3	60.5	74.9	58.3
Sparse AE	62.1	39.9	56.2	35.4
Sparse AE, mask-scaled	87.5	73.8	87.6	73.2
Sparse AE, mask-scaled, 2-pass	**88.4**	**73.9**	**88.1**	**75.6**

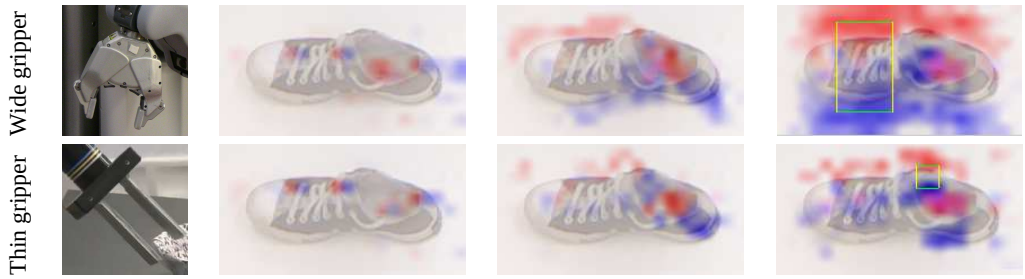


Figure 8: **Visualization of grasping scores for different grippers.** – Red indicates maximum score for a grasp with left gripper plane centered at each point, blue is similar for the right plate. Best-scoring rectangle shown in green/yellow. PR2's gripper (top) can span the entire shoe, and our system correctly detects vertical grasps along its length. The Viper gripper (bottom) cannot span the entire shoe, and thus strong grasps are only detected along the rim.

shows detection scores for systems constrained based on the PR2 and Adept grippers. For grippers with different properties, such as multi-fingered or jamming grippers, our algorithm would be able to learn new features for detection given only data labeled for the desired gripper.

4.2 Two-pass Detection System.

We tested our two-pass system by training a network with 50 hidden units at the first and second layers. Learning and detection were performed in the same manner as with the full-size network, except that the top 100 rectangles for each image were recorded, then re-ranked using the full-size network to yield a single best-scoring rectangle. The number of rectangles the full-size network needed to evaluate was reduced by roughly a factor of 1000.

Using our two-pass approach increased detection performance up to 2% as compared to a single-pass network. In most cases, the top 100 rectangles from the first pass contained the top-ranked rectangle from an exhaustive search using the second-pass network, and thus results were unaffected.

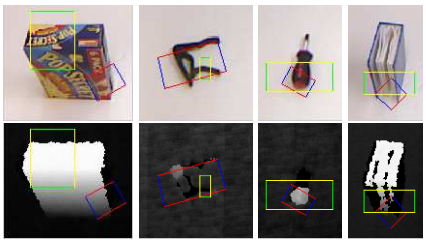


Figure 9: **Improvements from two-pass system.** – Example cases where the two-pass system produces a viable grasp (shown in green and yellow), while the one-pass system does not (shown in red and blue). Top: RGB image, bottom: depth channel

Figure 9 shows some cases where the first pass network pruned away rectangles corresponding to weak grasps which might otherwise be chosen by the second pass network. In these cases, the grasp chosen by the one-pass system might be feasible for a robotic gripper, but the rectangle chosen by the two-pass system represents a grasp which would clearly be successful.

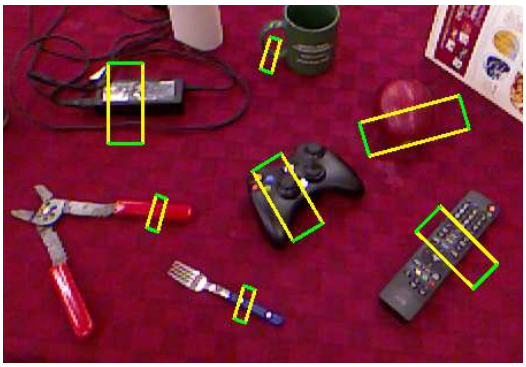


Figure 10: **Multiple grasp detection.** – A cluttered scene with many graspable objects for which our system identifies valid grasps.

The two-pass system also significantly increases the computational efficiency of our detection system. Average inference time for a MATLAB implementation of the deep network was reduced from 24.6s/image for an exhaustive search using the larger network to 13.5s/image using the two-pass system.

Finally, Figure 10 shows the result of our detection in a multi-object scenario. Given segmentation from the background, our algorithm is able to detect grasps for a wide variety of objects, even those such as the XBox controller which were not present in the dataset.

References

[1] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *JMLR*, 12:2493–2537, 2011.

[2] Advait Jain, Hai Nguyen, Mrinal Rath, Jason Okerman, and Charles C. Kemp. The Complex Structure of Simple Devices: A Survey of Trajectories and Forces that Open Doors and Drawers. In *BIOROB*, 2010.

[3] Yun Jiang, Stephen Moseson, and Ashutosh Saxena. Efficient grasping from rgbd images: Learning using a new rectangle representation. In *ICRA*, 2011.

[4] K. Kavukcuoglu, P. Sermanet, Y.L. Boureau, K. Gregor, M. Mathieu, and Y. LeCun. Learning convolutional feature hierarchies for visual recognition. *NIPS*, 2010.

[5] Ellen Klingbeil, Ashutosh Saxena, and Andrew Y. Ng. Learning to open new doors. In *RSS*, 2010.

[6] Alex Krizhevsky, Ilya Sutskever, and Geoff Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*. 2012.

[7] Quoc Le, Marc'Aurelio Ranzato, Rajat Monga, Matthieu Devin, Kai Chen, Greg Corrado, Jeff Dean, and Andrew Ng. Building high-level features using large scale unsupervised learning. In *ICML*, 2012.

[8] Quoc V. Le, Alexandre Karpenko, Jiquan Ngiam, and Andrew Y. Ng. Ica with reconstruction cost for efficient overcomplete feature learning. In *NIPS*, 2011.

[9] Honglak Lee, Yan Largman, Peter Pham, and Andrew Y. Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. In *NIPS*, 2009.

[10] G. Mesnil, Y. Dauphin, X. Glorot, S. Rifai, Y. Bengio, I. Goodfellow, E. Lavoie, X. Muller, G. Desjardins, D. Warde-Farley, et al. Unsupervised and transfer learning challenge: a deep learning approach. In *ICML*, 2011.

[11] A.R. Mohamed, T.N. Sainath, G. Dahl, B. Ramabhadran, G.E. Hinton, and M.A. Picheny. Deep belief networks using discriminative features for phone recognition. In *ICASSP*, 2011.

[12] M. Osadchy, Y. LeCun, and M.L. Miller. Synergistic face detection and pose estimation with energy-based models. *JMLR*, 8:1197–1215, 2007.

[13] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (fpfh) for 3d registration. In *ICRA*, 2009.

[14] Ashutosh Saxena, Justin Driemeyer, and Andrew Y. Ng. Robotic grasping of novel objects using vision. *IJRR*, 27(2):157–173, 2008. ISSN 0278-3649.

[15] Ashutosh Saxena, Lawson L. S. Wong, and Andrew Y. Ng. Learning grasp strategies with partial shape information. In *AAAI*, 2008.

[16] Kihyuk Sohn, Dae Yon Jung, Honglak Lee, and Alfred Hero III. Efficient learning of sparse, distributed, convolutional feature representations for object recognition. In *ICCV*, 2011.

[17] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, 2001.